Please amend the paragraph on page 2 beginning on line 5 as follows:

A2

Hyperlinks are typically inserted into documents manually or automatically. Manual insertion often occurs through a document editor or word processing program, such as Microsoft Word 6.0 or Corel WordPerfect 8.0, that includes a hyperlink definition capability. More particularly, manual insertion requires a user to select text in a document, to indicate a desire to associate a hyperlink with the text, and then to enter the URL for the document to hyperlink to. For example, a user writing a paper on the American Revolution might select text referring to George Washington and insert the URL for a document providing a brief biography of George Washington. However, manual insertion of hyperlinks can be not only tedious and time-consuming, but also error prone.

Please amend the paragraph on page 2 beginning on line 13 as follows:

A3

Conventionally, one of few methods for [A]automatic insertion of hyperlinks [often occurs] is through execution of an index-generation program, which builds an alphabetical "back-of-the-book" type index for a document. For example, WebAnchor (TM) software from Iconovex Corporation of Bloomington, Minnesota, uses semantic (meaning-based) analysis to extract key words, phrases and ideas from one or more documents and then automatically builds an index that includes hyperlinks to those extracted words, phrases, and ideas in the documents. A user viewing the index can select a word, phrase, or idea in the index and then hyperlink to the specific portion of the document containing it. (See also U.S. Patent 5,708,825, entitled Automatic Summary Page Creation and Hyperlink Generation, filed May 26, 1995 and issued January 13, 1998.)

Please amend the paragraph on page 2 beginning on line 22 and continuing on page 3, ending at line 5 as follows:

A4

One problem in using conventional manually and automatically generated hyperlinks concerns their longevity. In particular, as documents containing hyperlinks age, many hyperlinks become out dated and ineffective because the documents they point to have been deleted, revised, or moved to other computers, or because the computer hosting a hyperlinked document no longer recognizes or understands one or more parts of the URL for the hyperlink. Whatever the reason,

attempting to execute or invoke these ineffective hyperlinks results in an error message, such as "file not found," being shown to the user instead of the desired document. Although the user can update or replace the URL associated with an ineffective hyperlink, this can be inconvenient and time consuming, particularly in documents with many hyperlinks.

Please amend the paragraph on page 4 beginning on line 3 as follows:

Another feature of the exemplary method defines the name of the common computer system in the hyperlinks based on the cost or time of executing the hyperlink. In particular, the exemplary method determines whether the hyperlinks are to include the name of a local computer system that includes a database, for example, a CD-ROM library, containing legal documents or the name of an external computer system including this information. Another feature of this implementation allows for the possibility that the local computer system lacks this information and redirects failed hyperlinks to [the desired document on] a local computer system to the external computer system, which is likely to have a more extensive database.

Please amend the paragraph on page 5 beginning on line 11 as follows:

[The term "database," includes any logical collection or arrangement of machine-readable data.]

Please amend the paragraph on page 7 beginning on line 10 as follows:

Figure 1 also shows that exemplary document 128 includes portions 128a, 128b, and 128c which have been marked, for example, visibly in a contrasting color or font, or in other ways, to signify its association with an existing hyperlink. Document 128 also includes hyperlinks 129a, 129b, and 129c that are associated logically with respective portions 128a, 128b, and 128c. Hyperlinks 129a, 129b, and 129c which are defined in accord with the present invention, can be activated (in accord with conventional techniques) for example, to create a network connection to respective searchable databases 17, 18, and 19 via computer network 14 and web server 16. (Although Figure 1 shows that the hyperlinks all point to server 14 in the exemplary embodiment, other embodiments of the invention generate hyperlinks that point to the same or multiple web servers.)

Please amend the paragraph on page 7 beginning on line 20 and continuing on page 8 ending on line 6 as follows:

A8

In the exemplary embodiment, computer network 14 is a wide-area network such as the Internet; however in other embodiments it is a [local application,] local-area network, or an [enthroned] ethernet. Server 16 is a web server, such as a Microsoft Internet Information Server 4.0 running on a network of several NT servers with Pentium class processors and extended memory and disk configurations. Though not shown explicitly in Figure 1, server 16 includes URL processing software in accord with the invention (as described below.) Databases 17, 18, and 19 can take on any number of forms on a variety of computer platforms. Moreover, databases 17, 18, and 19 include overlapping content in some embodiments to allow for more than one computer, like computer 12, to hyperlink simultaneously to multiple copies of the same document. Thus, domain server 14 includes software capabilities such as that described in U.S. Patent 5,644,720 entitled Interprocess Communications Interface for Managing Transactions Requests and issued July 1, 1997. (This patent is incorporated herein by reference.)

Please amend the paragraph on page 8 beginning on line 9 as follows:

A9

Figure 2 shows an exemplary architecture for marking-and-link-building software 127, specifically a component-based architecture including one or more objects. The objects expose their functionality and communicate with other objects using COM (Component Object Model) interfaces. However, for clarity many of these interfaces have been omitted from the figures. The exemplary software is tailored to find, mark, and build hyperlinks for legal citations. However, other embodiments of the software operate similarly on proper names of persons or places, or other identifiable document content, which can be isolated and identified by syntactic, formatting, contextual, semantic, or document markup information.

Please amend the paragraph on page 8 beginning on line 17 and continuing on page 9 ending on line 6 as follows:

A10

More particularly, exemplary software 127 includes an integration object 127a, a tokenizer object 127b, a content-finder object 127c, a publications object 127d, a publication database 127e, a phrases object 127f, a phrases database 127g, an options object 127h, and a link-builder object 127i. Integration object 127a functions to integrate the software as a tool into

document-processing software 126 according to conventional techniques. Integration object 127a takes data from an active document in an active edit window of document-processing software 126 and passes it to tokenizer object 127b, as a stream of text. (In one embodiment, Integration object actually passes the text to the content-finder object, which then in turn passes it to the tokenizer object) Integration object 127a is also responsible for creating and initializing several other objects, such as tokenizer object 127b and content-finder object 127c and link-builder object 127h. (In one embodiment, the integration object does not always create the tokenizer object. If the integration object does not specify to the citefinder object which tokenizer object to use, the citefind object will create and use the default text tokenizer. This is the case in the exemplary Word and WordPerfect implementations. In the HTML implementation, the integration object creates the HTML tokenizer and passes it into the citefinder object.)

Please amend the paragraph on page 9 beginning on line 17 and continuing on page 10 ending on line 4 as follows:

Tokenizer object 127b receives, buffers, and parses the stream of text into a collection of tokens, which it passes to content-finder object 127c. In doing so, tokenizer object 127b, which assumes a variety forms depending on the actual form of the text it receives for tokenization, insulates content-finder object 127c from the actual format of the text, that is, it removes fonts and other features of the text that are deemed to carry no distinguishing value in locating legal citations. (However, in other embodiments, this formatting may carry useful semantic information and thus is not removed.) In one embodiment, software 127 includes several tokenizers, for example, an HTML tokenizer, a Microsoft Word tokenizer, a WordPerfect tokenizer, an Adobe Acrobat tokenizer, a text tokenizer, an RTF tokenizer, an XML tokenizer, a Microsoft Word Format tokenizer, a WordPerfect Format Tokenizer, and an Adobe PDF format tokenizer. In [muli] multi-tokenizer embodiments, document-processing software 127 selects or defines which one is necessary for the applicable text. If no specific tokenizer is specified, software 127 uses a default tokenizer object, which parses and creates tokens from straight UNICODE text.

Please amend the paragraph on page 10 beginning on line 13 and as follows:

A12

Each token created by tokenizer object 127b contains information that identifies what the entity is and where it is located in the document. For example, a text character is considered a single token. As another example, consider the HTML tag for line break
. An HTML implementation of the tokenizer treats this tag as a single token that represents a new line, even though it is four characters long. So the content-finder object only has to deal with a single new line token and does not have to [worry] "worry" that it is represented as
 in one format and "0x0d" in another.

Please amend the paragraph on page 11 beginning on line 6 and continuing on page 12 ending on line 18 and as follows:

A13

Content-finder options object 127h provides a mechanism for the content-finder client (document-processing software 126) to control or adjust properties of the cite-finding process. The client can indicate what types of cites they want to locate, as well as setting other options that control how the cites are located. The following list identifies and describes options available in the exemplary embodiment and other embodiments of the invention:

FindCaselaw	Indicates whether caselaw authorities should be located
FindStatutes	Indicates whether statute authorities should be located
FindLawReviewAndJournals	Indicates whether law review and journal authorities should be located
FindCourtRules	Indicates whether court rule and order authorities should be located
FindAdmins	Indicates whether administrative report and decision authorities should be located
FindRegulations	Indicates whether regulation authorities should be located
FindShortForms	Indicates if short form citations should be located
FindTitles	Indicates if titles should be found
FindInQuotes	Indicates if authorities within quotations should be located
OverlapSize	Amount of previous buffer saved when new buffer is passed in
MaxTitleLength	Maximum length of a title

MaxNumberLength	Maximum length of a number
MaxEditorPhraseLength	Maximum length of editor phrase
MaxKeywordPhraseLength	Maximum length of a keyword phrase
MaxStatuteKeywords	Maximum number of words examined when locating statute keywords
MaxStatutePubWords	Maximum number of publication words allowed in a statute authority
MaxCourtLength	Maximum length of court name
MinCourtLength	Minumum length of court name
MaxDateLength	Maximum length of a date
MaxExtensionPageDiff	Maximum distance allowed between page numbers considered to be part of same authority
BeginningYear	Specifies first number recognized as a year
EndingYear	Specifies last number recognized as a year

A13

In the exemplary embodiment, these options cannot be set by the user, although other [embodiment] embodiments allow this. The properties and definitions that can be changed (see note in next paragraph) are the same as the list of options above. The property is the thing (like Beginning Year); the method is the way to change the property.(See Table above.)

Please amend the paragraph on page 22 beginning on line 7 as follows:

A14

In block 318, the processor attempts to identify all statute-like citations. Similar to the case law processor, the statute processor examines all of the unused numbers in the number list. (Numbers are marked of as used in checking for short forms and case law.) Because case law processing is done first, many of the numbers in the number list will already be included as part of a citation and marked as used in the list. The statute processor skips all of the numbers that are marked as used and processing those that have not been marked [according] as follows.

Please amend the paragraph on page 23 beginning on line 22 and continuing on page 24 ending on line 11 as follows:

A15

[In block 322, content-finder object 127c notifies the client, that is, document-processing software 126, through integration object 127a about each of the found citations listed in the cite list. More particularly, the content-finder object creates a found object 127i (also referred to as a found object in the exemplary embodiment) for each listed cite, using the information in the cite list entry to fill in the properties of the found object. The content-finder object then triggers a cite-found event that is captured by the client, with the found object specified as a parameter of the cite-found event. After all of the cites have been returned to the client, the cite and number lists are cleared. In block 324, the processor, through integration object 127a, marks each of the found cites in the document. In the exemplary embodiment, this entails getting the citation position from each of the found objects and locating the citation in the original text of the current document. Integration object 127a then marks the citation (or found text) with the appropriate hyperlink. First, it examines the text in the range of the citation to determine if any hyperlinks already exist. Then, depending on an overwrite-existing-hyperlinks option (defined in the cite-link options object), it will either overwrite the existing hyperlink(s) or it will not insert the conflicting hyperlink.]

In block 322, content-finder object 127c notifies the client, that is, document-processing software 126, through integration object 127a about each of the found citations listed in the cite list. More particularly, the content-finder object creates a found object 127i (also referred to as a found object in the exemplary embodiment) for each listed cite, using the information in the cite list entry to fill in the properties of the found object. The content-finder object then triggers a cite-found event that is captured by the client, with the found object specified as a parameter of the cite-found event. After all of the cites have been returned to the client, the cite and number lists are cleared.

In block 324, the processor, through integration object 127a, marks each of the found cites in the document. In the exemplary embodiment, this entails getting the citation position from each of the found objects and locating the citation in the original text of the current document. Integration object 127a then marks the citation (or found text) with the appropriate hyperlink. First, it examines the text in the range of the citation to determine if any hyperlinks already exist. Then, depending on an overwrite-existing-hyperlinks option (defined in the cite-

A15

link options object), it will either overwrite the existing hyperlink(s) or it will not insert the conflicting hyperlink.

Please amend the paragraph on page 25 beginning on line 1 as follows:

A16

After completion of block 328, the exemplary method continues optionally with activities related to generating a Table of Authorities or other foundcontent collection and collation activities (e.g. creation of an index or Table of Contents). In particular, if the user has selected to generate a Table of Authorities (TOA), the integration object will insert the appropriate TOA entry tag for the citation. This entails determining if the citation already has a TOA entry tag. If there is an entry tag for the citation, the Integration object removes it and instructs the client application to insert a TOA entry tag immediately following the citation. How this is accomplished is dependent on the client application. The type of TOA entry tag is also dependent on the whether or not the[foundcitation] found citation is a long form or short form. Once all the TOA entry tags have been inserted, the user can select to generate a Table of Authorities in accord with the particular document-processing software.

Please amend the paragraph on page 25 beginning on line 13 as follows:

A17

In the exemplary embodiment, link-builder object 127i applies a consistent URL syntax based on the Internet URL RFC 1738 (which is incorporated herein by reference.) A seminal aspect of the structure is that rather than including a specific filename or location for a document, it includes information that a web server, such as server 16 in Figure 1, can process to find the document with high certainty. In the exemplary embodiment, the content is assumed to be a legal citation and the hyperlink is built to direct the hyperlink to a server that provides access to legal documents. However, in other embodiments, it can be specific content understood to refer to scientific or academic citations or classes of terms. Even more broadly, one could simply direct hyperlinks to semantically key terms in a document to a common universal data provider. In any case, a major advantage of the predefined structure for automatically generated hyperlinks is that changing the file name or file location within (and in some embodiments without) the domain does not invalidate any hyperlinks referencing the document.

Please amend the paragraph on page 26 beginning on line 9 as follows:

A18

Examples for domain-name field [404] 402 include www.keycite.com, www.westlaw.com, and www.westdoc.com. Examples of application-specific-path information field [406] 404 are subdirectories or electronic commerce token information. Thus, for example, one embodiment of the structure includes customer account information for the particular domain name or a credit or debit account number with associated personal-identification numbers. These can be specified within a computer systems, like system 12 in Figure 1, and inserted by link-builder object 127i into a particular hyperlink.

Please amend the paragraph on page 31 beginning on line 1 as follows:

A19

In these examples, the request source must be registered with content provider. The "&sp= parameter" identifies the sponsor for the request. The operation-specific-parameters (that is, "&db=, &method=, &query=, &action=") are required fields to retrieve a specific search result. [If you omit] With omission of this parameter, one will gain access to the service, but will not retrieve a specific result. In the exemplary embodiment, the "&db= parameter" must be a valid database identifier code. A list of valid database identifiers must be obtained from the content provider. The valid search methods for the "&method= parameter" are **tnc** for terms and connectors and **win** for natural language.

Please amend the paragraph on page 32 beginning on line 12 as follows:

A20

Once a hyperlink, such as hyperlink 129a, 129b, or 129c (in Figure 1), is selected or invoked by a user, computer 12 establishes a connection via communications device 122 and network 14 to server 16. Server 16, which serves the domain name identified in the hyperlink, forms an HTML page that may include the following information:

Please amend the paragraph on page 32 beginning on line 16 as follows:

A21

Users invoking a hyperlink can be asked for a username and/or password for the first access to a service. Subsequent accesses through the same service through the same or different hyperlinks to the same service will not (in most cases) result in another authentication prompt. However, if too much time has elapsed, for example three months, since the last request[.], authentication is prompted. The amount of time allowable between requests may vary for each service.

Please amend the paragraph on page 33 beginning on line 2 as follows:

A22

Additionally, the content for a specific citation can change over time. For specific citations, most services will return the most current version of a document rather than an older version. For this reason, a cite retrieved for an older hyperlink can return a newer version of the document if the same link is run in the future. Similarly, the same search completed through a given hyperlink can provide time variant results because more documents may meet [it] its encapsulated search criteria.

Please amend the paragraph on page 33 beginning on line 8 as follows:

A23

Because of the persistent nature of URLs, the use of any authentication, or user-specific information in the URL is avoided in the exemplary embodiment. While it is likely that the content addressed by this proposal will be protected and the user will need to present authentication information, the prompting for this [remain] remains outside of the actual URL syntax in the exemplary embodiment. Users, however, can share URLs without allowing others access to their data access account.

Please amend the paragraph on page 33 beginning on line 14 as follows:

A24

One potential use of the invention would be when publishers may want to allow other users access to their account in the form of sponsored links. The third party publisher would have an agreement with the content provider to provide a slice of content to their subscribers or even to the general Internet community. In this case, pages at the publisher's site would contain URL links [to]that link to the content provider . These URL links would contain a sponsorship parameter ("&sp=") to identify the sponsoring publisher and any required authentication information. The use of the sponsor parameter does not preclude the use of other forms of authentication, since the various business rules are contained on the server.